目　　录

第一章：开发板简介……………………………………………...…3

1－1．SY_1126 开发板的特性简介………………………...…3

1－2．SY_1126 开发板的构成和工作原理……………….….4

第二章：开发板使用说明…………………..……………………..6

2－1. 使用简介及入门指导………………………………………6

2－2．键盘规划及编码规划，键盘功能，功能扩展……………7

2－3. 在线下载功能一(AT89S8252_AVR 并口另配下载器)……8

2－4．在线下载功能二(AT89S52xx)……...…………………….10

2－5．在线下载功能三(P89C51RD2xx)………………………..11

第三章：开发板用器件资料及说明……………………………13

3—1．AT89S8252……………………………………………...…13

3—2．AT89S51(2)………………………………………………...13

3—3．P89C51RD2xx……………………………………………14

3－4．AT24Cxx…………….…………………………………...14

3－5．DTLED-6……………………………………………….…16

第四章：开发板器件表附件清单………………………………19

4—1．调试用源程序………………………………………………19

4－2．原理图……………………………………………….附录插页

4－2．包装清单……………………………………………………42

第五章：其它５１类实验板简介………………………………..43

5－1. 51DEMO I/O 板简介……………………………...………43

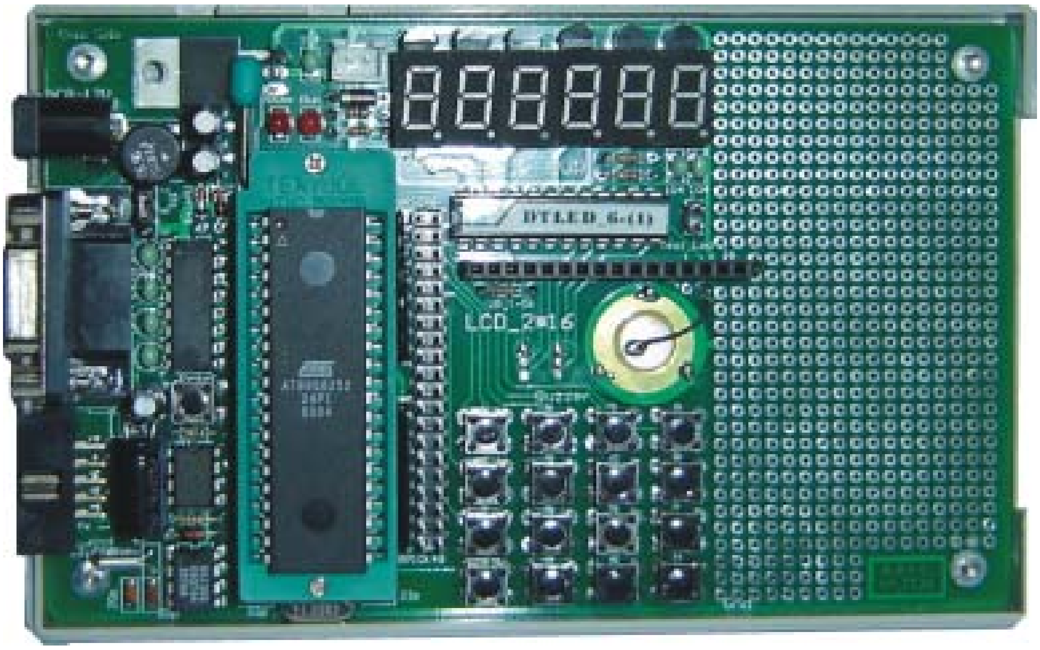
5－2．89C51 数模转换实验板简介.....................................44
5－3．流水灯控制器（12 路）.......................................45
5－4. ISP 下载线（选配自购件）.....................................46

**********公司其它产品简介见软件盘中电子版文件**********

第一章：开发板简介

1－1．SY_1126 开发板的特性简介

开发板实物示意图 ：



下载连接器实物示意图 ：

专用下载器[正面]

专用下载器(反面)

1． 基本功能与 SY-0606 开发板相同。

2． 全新支持 89S8252 和 AVR 系列芯片的下载应用。

3． 标准的 8X5X 应用电路设计，电源/时基/复位/端口驱动。

4． 自带程序的在线烧录（自下载）功能。

5． 提供 16*2 字符显示液晶板接口(液晶模块需另选配)。

6． 带标准 RS232 接口连接电路。

7． 有掉电密码不消失之功能串行 EEPROM 应用。

8． 自带 4*4 标准键盘输入，便于学习者掌握键盘输入和程序编写。

9． 精确标准的时钟电路，（另加备用电源）可直接作为时钟计时用。

10．用串行驱动方式，驱动 6 位数码管显示，大大节省了单片机的接口资源（祥见后面“ DTLED-6 ”芯片介绍）。

11． 有一对蜂鸣器驱动输出在显示芯片上，不占用单片机的口资源。

12．预留扩展空间及接口，可直接驱动或控制用户设备，便于用户直接开发品。

1－2．SY_1126 开发板的构成和工作原理

1. 在采用了 Atmeal 公司的“ AT89S8252 ”等 AVR 系列芯片时，另配上图所示的“ AVR 下载器 ”（选配件），用户可以直接在正常工作的模式下，通过板上 ISP 口进行代码下载，将用户在 PC 机上的程序代码，通过计算机的并行口（打印口）方式下载到单片机板上直接演示，特点是简单方便，性价比高。是单片机初学者的首选。

2. 在采用 PHILPS 公司的“ P89C51RD2 ”芯片时，用户可以通过板上的 J1 跳线设置，直接通过 RS232 口将用户在 PC 机上的程序代码下载到单片机上直接演示，免去再购置烧录器的重复投资。

3.　　本产品集单片机的最小系统应用于一体，在端口的应用设计上进行了优化改进，力求简洁明了，资源共享开放，方便实用；为单片机的二次开发和初学者提供极大的方便。其电路中包含典型的电源整流稳压电路，复位启动电路，晶体振荡时钟电路，键盘输入电路，RS232 串行通讯电路，串行 EEPROM 的在线擦写（可随时更改信息密码），LED 数码管扫描显示电路等，精确时钟读写等，是单片机初学者的精典范例。也是工程开发人员可随手取及的最佳半成品。

工作原理： SY_1126 开发板是原 SY_0606 板的升级更新产品，仍有像其它 51 类实验板一样的功能，上电复位后进入启动工作状态，程序自动将 LED 数码管上电显示清零，并自动进入时钟计时状态，蜂鸣器会有 5 秒钟的提示音响，进入等待接收键盘指令状态；当在键盘上输入六位数字的数码时，单片机通过键盘输入口（即定为 P1 口），将键盘的键值读入处理后，将其对应值，通过显示芯片传送数据到 LED 数码管显示出。

　　当接收到键盘上“ D ”键开锁信号时，单片机会自动判断所收到的键盘码（目前数码管上的数字）是否与内存密码相同？如果判断“ 是 ”开锁信号（键盘码与内存密码相同），则将锁打开（绿色下载指示灯亮，有 5 秒钟蜂鸣器提示音响），表示给出了一个开锁动作的信号，此时程序自动进入时钟设置修改和串口信号接收状态，时种修改可通过“ 加/减/左移/右移 ”键（2 / 8/ 4/ 6 键）进行，设置完成后请按一下“ * ”储存键来修改当前时钟。在开锁状态下还可接收串口传来的十六进制数据（如 01 02

03 04 05 06）并在数码管上显示出（123456）来；如果判断“ 不是 ”开锁信号（键盘码与内存密码不同），系统则不执行此指令。

当接收到键盘上“ * ”键存储密码信号时，单片机会自动将所收到的键盘码（目前数码管上的数字）储存于 Eeprom(24C02)中，作为本板的开锁密码，以备下次开锁时使用。Eeprom 是掉电存储密码的设备，也就是说当密码设定后，开发板即使掉电后密码也不会消失；若键入的数值均非单片机预设的操作指令码，则本单片机系统不预执行它。

当接收到键盘上“ C* ”键显示存储密码信号时，单片机会自动将储存于 Eeprom(24C02)中已设密码在数码管上显示出来。

4*4 键盘输入部分与 89X8252 单片机的 P1 口相连接，也是以扫描的方式来读取键位值；单片机初始内设定密码为“ 111111 ”，内定密码只是提供给未设定密码的新板产品。

通用串口的连接，在初始运行状态（）下，会将当前的显示时间，通过串口发送到 PC 机上，供终端设备显示用。在开锁状态下，可接收从 PC 口传来的十六进制数据数字，并将其收到的数字在 LED 上显示出来。

而 P0 口是与 16*2 液晶显示板的数据口相连。在单片机启动或按“ 复位 RESET ”键后，会自动显示“ Syber Nanjing.Co ”和“ Password: xxxxxx ”的字样。

第二章：开发板使用说明

2－1．使用简介及入门指导

将 SY_1126 开发板接上电源 AC8V~12V（或 DC+5V）。电源接通后，红色的电源指示灯点亮，数码管显示应为初始时钟“ 0000.00 ”，（加上备用电池后就可以显示系统里的已设标准时间）。进入计时状态系统正常。

显示已存密码： 按住“ C ”键即可读到已存的密码，初始密码由 8X52 程序中的“ TAB[111111] ”决定，所以初值为“ 111111 ”的密码，或从 Eeprom 中读取上一次已更改的密码；放开则恢复显示时钟 “ 0000.00 ”。

清除输入显示： 按“ # ”键为清除显示器，数码管显示即为“ 0000.00 ”。

设置（更改）密码： 欲更改密码时，先键入新密码，然后按“ * ”键，即可建立新密码，并存入 Eeprom(24C02)的地址中，以备下次启动时用。

进入时钟调整设置：

请在已开锁状态下进行：

按“ 4 ”键（左移键）：左移当前调整位。

按“ 6 ”键（右移键）：右移当前调整位。

按“ 2 ”键（加一键）：当前调整位数据加一。

按“ 8 ”键（减一键）：当前调整位数据减一。

调整完成后，按“ * ”键（存储键）将修改后的时钟值存入

DS1302 时钟芯片中。(加上备用电池后就可以显示系统里的已设标准时间)。

如要回到初始密码锁状态（不修改时钟）只需按一下“ A ”键或复位键（Reset）系统重新启动即可。

2－2．键盘规划及编码规划，键盘功能，功能扩展

（1） 键盘规划如图：

（2） 编码规划如图：

01	02	03	0C
04	05	06	0D
07	08	09	0E
0A	00	0B	0F

（3） 键盘功能

1． “ 1--9 ” 数字键为输入数字键，输入密码有效位每次为 6 位数字；在开锁状态下，4 为左移，6 为右移，2 为加一，8 为减一，

2． “ * ” 符号键，为密码输入的确认键。

3． “ # ” 符号键，为显示屏清除键。按下后清除数码管

为“ 000000 ”。

4． “ D ”符号键，为开锁确认键。密码输入正确后按下此键，既可打开驱动，绿色灯闪亮或喇叭响作为指示。

5． “ C ”符号键，为密码显示键。按下后即可显示已存储在板内的开机密码。

6． “ A ”符号键，为恢复显示键。按下后即可恢复到时钟计时显示。

7． “ B ”符号键，为机动保留键。用户可在源程序中设定其功能使用。

2－3．在线下载功能一(AT89S8252xx 并口另配下载器)

1. 将光盘上 SLISP 软件正确安装完成后。
2. 将下载板上的功能跳线 J1 为 1-2 短路的运行状态。
3. 将另选配的 AVR 下载器电缆的一端插在本 SY_1126 板上的 ISP 接口上，另一端插在下载器的接口上，然后再将下载器接到 PC 机的并行口（打印机端口）上。
4. 再按图示极性将+5V 电源接在本开发板上，此时板上红色 Power 指示灯亮。
5. 运行 SLISP 软件，出现如图所示画面。

进行下载（烧录）设置

按上图正确选择通讯端口。

选择烧录的 IC 芯片的型号：AT89S8252

按正确路径将烧录的程序代码设在 FLASH 存储器中。

按“ 重载 ”键后，即可装载烧录代码，按图示命令进行下载烧录芯片。在操作时板上的绿色运行指示灯会闪亮。

将开发板复位后即可进入单片机的自运行状态。

2－4．在线下载功能的使用二（**P89S51/2/5**）：

按上述同样的使用方法进行操作。

选择烧录的 IC 芯片的型号改为 AT89S52 即可。

2－5．在线下载功能的使用三（P89C51RD2）：

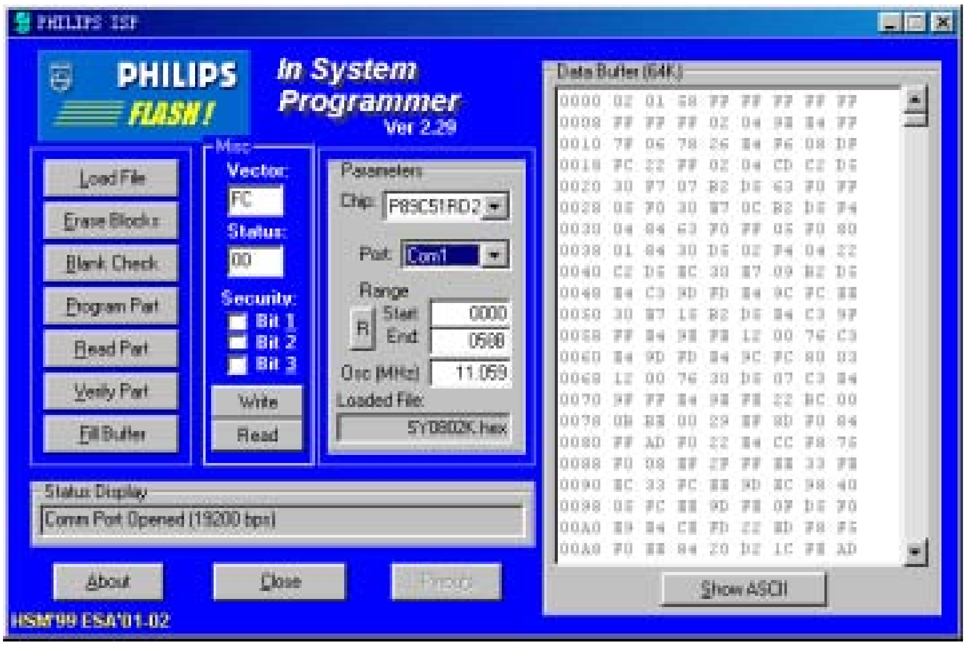
1. 将光盘上 Winisp 软件正确安装完成后。

2. 将本开发板的串口与 PC 机串口直接连接。

3. 将开发板上的功能跳线 J1 设为 2-3 短路（1-2 短路为运行，2-3 短路为下载）

4. 按图示极性将电源（+5V）加在本开发板上，此时红色 Power 指示灯亮。在下载运行时绿色运行信号灯会闪烁。

5. 运行 WINISP 软件，出现如图所示画面。



进行下载（烧录）设置

选择烧录的 IC 芯片的型号：P89C51RD2

选择通讯连接口：COM1-------COM4

选择晶体振荡频率：本电路板上是 11.059Mhz

设置芯片状态设定：Status=00，Vector=FC（只要在第一次烧录时设定一次即可）

写入芯片状态的设定值：点击“ Write ”键，出现如下画面

时请选“ 是 ”。

在“ Load File ”下装入用户的程序代码(***.HEX 或***.BIN)

在“ Erase Blocks ” 下擦除芯片，可选择全擦除或部份擦除(如是新空芯片此项可免除)。

在“ Program Part ” 烧录（下载）已装入的用户的程序代码到 P89C51RD 中。

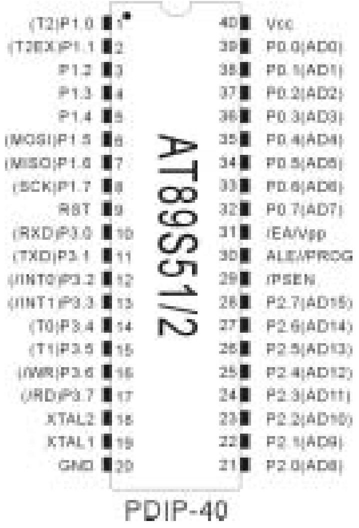
当出现“ Flash Programming ” 提示时，恭喜您已掌握了此类芯片的下载功能，烧录成功。

将开发板上的功能跳线 J1 转回设为 1-2 短路，复位后即可进入单片机的自运行。

第三章：开发板用器件资料及说明

3—1．AT89S8252

芯片平面引脚功能如图：同 AT89C51/2



- 标准的 MCS-51 指令模式
- 8K Bytes 的 Flash Memory
- 2K Bytes 的 Eeprom Memory
- 宽工作电压范围，可适应 4V—6V 的工作电压。
- 时钟频率可从直流到最高 24Mhz。
- 三重加密位，加密性能好。
- 256x8Bit 的内部 RAM。
- 有 32 个独立的 I/O 输入/输出端口。
- 三个 16 位可定义定时器。
- 一个可控制编程的串行通讯 UART 口。
- SPI 串行在线编程功能。
- 有看门狗功能。
-

3—2．P89C51RD2

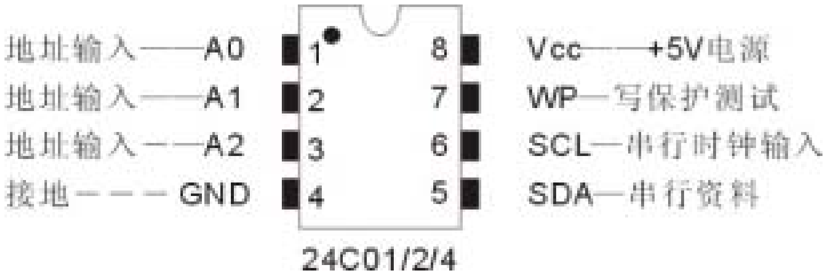
芯片平面引脚功能与上图 AT89C51 相同：(略)

PHILIPS 公司的 P89C51RD2 芯片具有并行可编程 62KB 非易失性存储器，可实现对器件的串行在线编程（ISP）和在应用中编程（IAP）；片内 ROM 中出厂时已固化有加载驱动程序，允许 ISP 通过 UART 将程序代码装如 FLASH 存储器中,而在用户的 FLASH 代码中则不需要加载程序，便于与 AT89C51 系列的兼容替代。

P89C51RD2 芯片是 6 个时钟周期为一个机器周期，因此，其运行速度是普通 51 芯片的二倍(设定可选择）;片内增加 64KB 的 EEPROM 和 1KB 的 RAM ;双 DPTR 指针和新增看门狗定时器；该器件是 8051 的派生产品指令系统与 80C51 完全相同。

3一3．AT24C02

芯片平面引脚功能如图：



对 24C01/2/4 的读写操作的方法和使用中应注意的问题。24C04 的 1一3 脚分别为器件编址端 A0、A1、A2 (用于与系统中的同类器件编码),4 脚为电源地,5 脚为 IIC 总线的数据线 SDA，6 脚为 I2C 总线的时钟 SCL，7 脚为测试输入端，在系统中接地，8 脚为电源。 要正确的对 24C04 进行读写操作首先要了解这两

个问题 1、器件寻址方法　2、翻卷现象及处理方法。器件的寻址方法：（24C 系列器件代码为 1010）。

R/W 位为读和写状态位，为 0 时写；为 1 时读。

电路中 IIC 器件采用的是 24C01，由于 24C01 的容量为 128 字节，如要对 24C01 进行读操作时，寻址字节就由：器件代码（1010）＋器件 A0、A1、A2 脚的状态（00）（状态与访问的单元有关，大于 256 字节时为 1 否则为 0）＋R/W（读 0/写 1）组成。

翻卷现象及处理方法：在此省略。本着让读者在不了解 IIC 的工作原理的情况下能拿来就用的原则，简单时序和使用方法如下：

A.指定地址写 N 个字节：

启动总线->发送器件寻址字节->回答->发送寻址地址->回答->发送第一个数据->回答->发送第二个数据->回答->....->发送第 n-1 个数据->回答->停止总线。

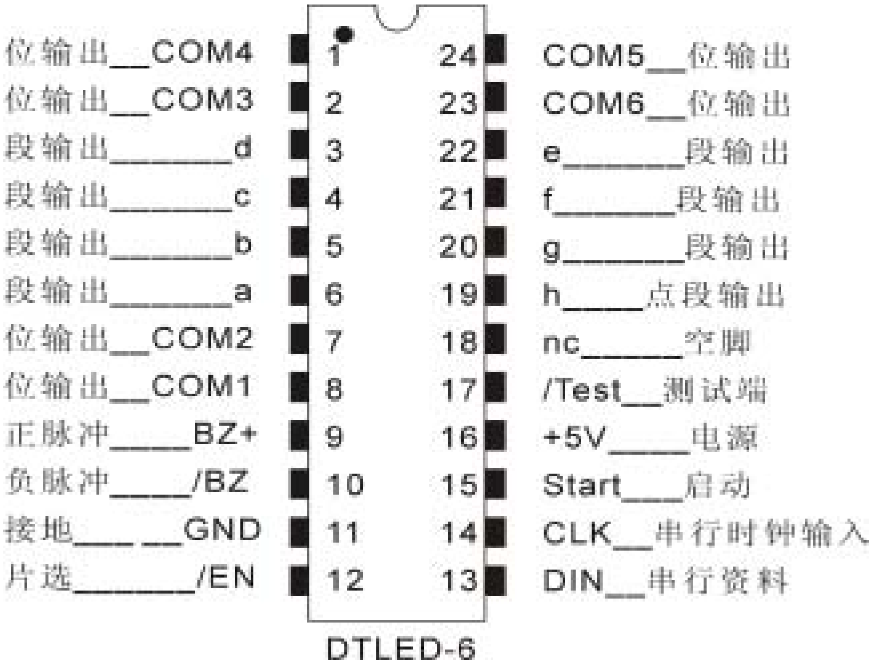
B.指定地址读 N 个字节：（数据寻址地址放在数据中的第一位字节。）

启动总线->发送器件寻址字节->回答->发送数据寻址地址->回答->停止总线+启动总线->读第一个数据->回答->读第二

个数据->回答->............->读第n-1个数据->回答->停止总线。

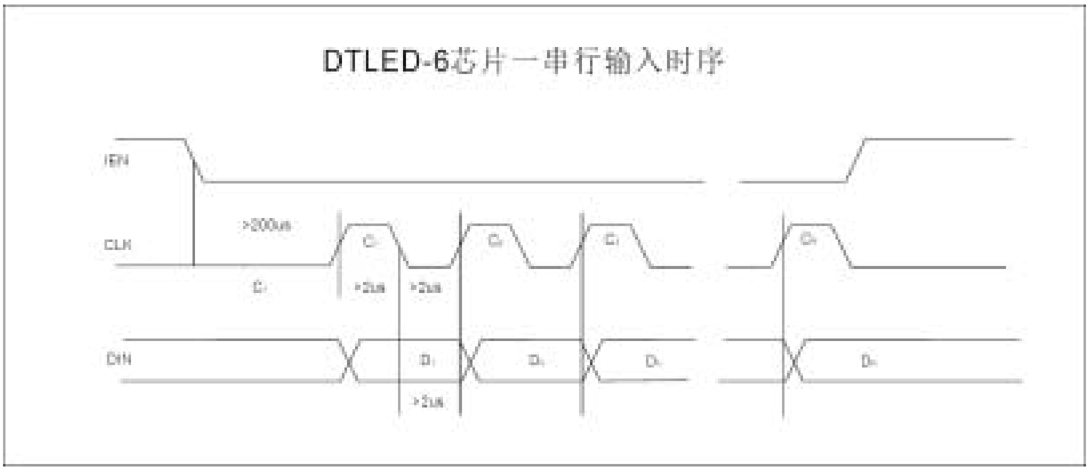
3－4．DTLED-6 串行接口 BCD 译码显示片

芯片平面引脚功能如图：



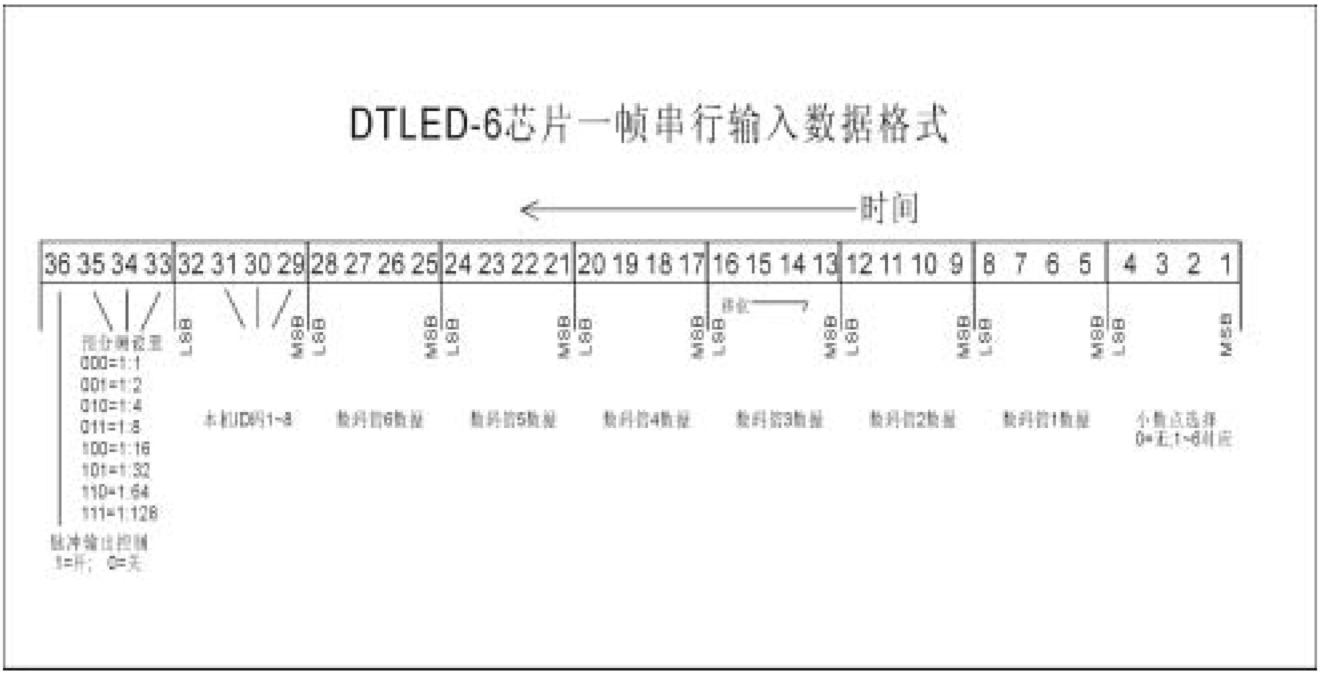
芯片上的 P17 脚为测试端。工作时为“ 1 ”电平;

当其为“ 0 ”电平时，P5/P4/P3/P2/P22/P21/P20/P19 为零电平，输出端接的数码管以次全亮，P9/P10 端有一对反向脉冲输出（蜂鸣器响）。



“ DTLED-6 ”芯片是在摩托罗拉的“ MC14499 ”芯片上延伸出来的（可参阅 MC14499 的使用），与“ MC14499 ”芯片的指令相同，并增加更多的功能，一片芯片可显示 6 个数码管，而且还可 N 片芯片并在一条线上，同时驱动多组 N*6 个数码管，自带芯片 ID 码识别功能，购买时可以选定 ID（本机中使用的 ID=01），市价只在几元钱，是同产品中的性价比极高的优选器件。由于 DTLED-6 片内具有 BCD 译码器和串行接口,所以它可以与任何单片机接口相连。DTLED-6 每一次可接收 36 位串行输入数据，32 位串行数据依序提供了 6 个数码管小数点的位选择，6 位数码管的 BCD 码，本机 ID 识别码，蜂鸣器分频输出码。

其串行输入的时序如图所示：



前 4 位为 0~6 选项，对应控制 6 个数码管的小数点是否显示，0 则不显示小数点。（软件中设为 3，对应于第三个数码管的小数点亮。）

5~28 位（4*6 位）是 6 个数码管显示值输入数据其相应的字符如表所示：BCD 码显示字符表

0000	0	1000	8
0001	1	1001	9
0010	2	1010	A
0011	3	1011	B
0100	4	1100	C
0101	5	1101	D
0110	6	1110	E
0111	7	1111	F

29~32 位是本芯片的识别 ID 码，用于在多片芯片并联使用时的识别检查。（软件中设为 1，选用“ DTLED-6 ”芯片时选用 ID=1 的即可。）

32~36 位是一对蜂鸣器输出控制和预分频输出设置。36 位=1 时，蜂鸣器输出开；36 位=0 时，蜂鸣器输出关；35/34/33 组成 0~7 级的预分频数。

第四章：开发板器件表附件清单

４—1．调试用源程序：

```
//*****************************************************************
//***此程序为调试通过的源程序
//***由南京赛博电子有限公司提供
//***文件名:SY1126.C***//
//***是 51 系例单片机的最小系统
//***有 AT89S8252/AT89S51/52 的在线下载功能
//***IIC 总线芯片为 24C02,记存密码
//*****************************************************************
#include"at89x52.h"
#include<intrins.h>
#define    uchar unsigned char
#define    uint    unsigned int
#define    nop    _nop_()           /*  定义空操作指令  */
#define VAR        P1
sbit clk=P2^0;                /*DTLED-6 显示芯片/时钟芯片 DS1302 时钟输入端*/
sbit dout=P2^1;               /*DTLED-6 显示芯片数据读入端*/
sbit en =P2^2;                /*DTLED-6 显示芯片使能端*/
sbit RST=P2^3;                /*DS1302 芯片复位起始端*/
sbit DINOUT=P2^4;             /*DS1302 时钟芯片数据输入输出端*/
sbit enable=P2^5;             /*LCD 模块用*/
sbit rw=P2^6;                 /*LCD 模块用*/
sbit rs=P2^7;                  /*LCD 模块用*/

sbit SCL=P3^4;                /*24C01 CSL 接脚=89C51 T0 P3.4*/
sbit SDA=P3^5;                /*24C01 SDA 接脚=89C51 T1 P3.5*/

bit FLAG0=0;                  /*位标号 FLAG0=1 键盘扫描回应*/
bit FLAG1=1;                  /*位标号 20H.1 比较密码回应旗号*/
bit FLAG2=1;                  /*位标号 FLAG2=0 时，进入键盘操作*/
bit FLAG3=1;                  /*位标号 FLAG3=0 时，键盘操作*/
bit FLAG4=0;                  /*位标号 FLAG4=0 时，键盘操作*/
bit FLAG5=0;                  /*位标号 FLAG5=1 时，存储*键按下*/
```

```
bit FLAG6=0;                     /*位标号 FLAG6=1 时，有串行信号输入*/
bit TSendAddress(uchar slaaddress,uchar subaddress,uchar * s,uchar no); /* 向有子地址器件写入 6 字节数据函数 */
bit TRcvAddress(uchar slaaddress,uchar subaddress,uchar * s,uchar no);  /* 向有子地址器件读取 6 字节数据函数 */
bit acknow;                      /* 应答标志位 acknow=1 表示正常响应 acknow=0 表示未响应 */
static const char tab[16]={0x01,0x02,0x03,0x0c,         /*键盘码 00,01,02,03,*/
                           0x04,0x05,0x06,0x0d,         /*键盘码 04,05,06,07,*/
                           0x07,0x08,0x09,0x0e,         /*键盘码 08,09,0a,0b,*/
                           0x0a,0x00,0x0b,0x0f};        /*键盘码 0c,0d,0e,0f*/
//static const char tab1[6]={0x01,0x01,0x01,0x01,0x01,0x01};    /*内定密码"111111"*/
uchar char1[]="Syber Nanjing.Co";
uchar char2[]="Password:";
uchar dispbuf[9];                 /*显示值存放阵列*/
uchar clocktmp[3];                /*显示值存放阵列*/
uchar bufdata[9];
uchar bufuart[6];                 /*串口接收值存放阵列*/
uchar a1=0,clockbak,b1=100,stand=0,m=0,busy,count=10;
char ww=0,ptr=0,ptr1=0;           /*PTR 键盘扫描指标，ptr1 显示器扫描指标*/
void delay (unsigned int value)   /*延时副程式*/
  {
      while (value!=0) value--;   /*10US 延时*/
  }
void Enable(uchar c);
void Write1(char *c1);
void Write2(char c2);
void Write3(void);                /* 显示密码字符串 */
void Initial();                   /*LCD 模块初始化*/
void tsled(void);                 /*串行发送到 DTLED-6 副程式*/
void BCD(void);                   /*BCD 码转换副程式*/
void scan(void);                  /*键盘扫描副程式*/
void clear(void);                 /*清除按键存放/显示器阵列 dispbuf[]副副程式*/
void open(void);                  /*开门比较密码副程式*/
void read24c02();                 /*读 24c02 的存储值*/
void write24c02();                /*写 24c02 的存储值*/
void IICSAVE(void);               /*设定密码并存入 buft[]阵列副程式*/
```

```
void disp(void);                    /*显示存放在 dispbuf[]阵列的密码副程式*/
void tserial(void);                 /*串口 serial 发送副程式*/
void rserial(void);                 /*串口 serial 接收副程式*/
void temp(void);
void Rd1302st(void);                /*读时钟芯片 DS1302*/
void Rd1302(void);                  /*读时钟芯片 DS1302*/
void Wr1302(void);                  /*写时钟芯片 DS1302*/
void Start();                       /*  起动总线函数  */
void Stop();                        /*  结束总线函数  */
void SendByte(uchar c);             /* 8951 发数据或地址给 8583  字节数据发送函数  */
uchar RcvByte();                    /* 8951 从 24C01 读数据字节数据接收函数  */
void Ack(bit a);                    /*  主机 8951 应答子函数  */
void xch(void);                     /*宣告按键存放/显示器阵列 dispbuf[]右键滚入副程式*/
void wxb1(void);                    /*0~9 计数,秒*/
void wxb2(void);                    /*0~9 计数,分*/
void wxb3(void);                    /*0~9 计数,时*/
void save_clocks(void);             /*按显示时间更改时钟*/
void speekclose(void);
void LCD_disple(void);              /*液晶显示*/
/**********************************************************************/
main()                    /*主程式*/
{
    TMOD=0x11;            /*TIMER0 工作在案 MODE1,16 位定时器*/
                          /*TIMER1 工作在 MODE2,自动重新装载模式*/
                          /*方式寄存器 TMOD____________________________*/
                          /*|       定时器 1           |        定时器 1       |*/
                          /*| GATE|C/T| M1 | M0 | GATE|C/T| M1 | M0 |*/
                          /*|____________________|_________________|*/
    TH1=256-(28800/9600);  /*设定传输波特率 9600，晶体振荡器=11.059Mhz*/
                          /*|波特率 9600 时|SMOD=0|C/T=0|模式=2|自动载入 TH1=FD|*/
                          /*|波特率 4800 时|SMOD=0|C/T=0|模式=2|自动载入 TH1=FA|*/
                          /*|波特率 2400 时|SMOD=0|C/T=0|模式=2|自动载入 TH1=F4|*/
                          /*|波特率 1200 时|SMOD=0|C/T=0|模式=2|自动载入 TH1=E8|*/
     TR1=1;               /*启动 TEMER0*/
//  T2MOD=0x02;              /*定时器 2 工作在输出模式*/
    T2CON=0x38;              /*定时器 2 工作控制寄存器 T2CON__________________*/
```

```
                        /*| TF2|EXF2|RCLK|TCLK|EXEN2|TR2|C/T2|CP/RL2|*/
                        /*|____________________|______________________|*/
    TL2=0xfd;           /*设定传输波特率 9600，晶体振荡器=11.059Mhz*/
    TH2=0xff;           /*设定传输波特率 9600，晶体振荡器=11.059Mhz*/
    RCAP2L=0xdc;        /*设定传输波特率 9600，晶体振荡器=11.059Mhz*/
    RCAP2H=0xff;        /*设定传输波特率 9600，晶体振荡器=11.059Mhz*/
    TH2=256-(28800/9600);  /*设定传输波特率 9600，晶体振荡器=11.059Mhz*/
                        /*|波特率 9600 时|SMOD=0|C/T=0|模式=2|自动载入 TH1=FD|*/
                        /*|波特率 4800 时|SMOD=0|C/T=0|模式=2|自动载入 TH1=FA|*/
                        /*|波特率 2400 时|SMOD=0|C/T=0|模式=2|自动载入 TH1=F4|*/
                        /*|波特率 1200 时|SMOD=0|C/T=0|模式=2|自动载入 TH1=E8|*/
     TR2=1;             /*启动 TEMER2*/
    SCON=0x50;             /*设定 UART 工作在 MODE3 模式，可传送和接收数据  REN=1*/
                        /*串列控制寄存器 SCON________________________*/
                        /*|   7      6      5      4 |   3      2      1      0 |*/
                        /*| SM0| SM1| SM2| REN| TB8| RB8|   TI |   RI |*/
                        /*|____________________|____________________|*/
     IE=0x98;           /*8 是允许中断，a 是 TIMER0，TIMER1 中断致能位*/
                        /*|   7    6    5    4 |   3    2      1      0 |*/
                        /*| EA |      | ET2| ES | ET1| EX1| ET0| EX0|*/
                        /*|_________________|__________________|*/
     IP=0x18;           /*中断优先级寄存器 IP________________________*/
                        /*|      |      | PT2| PS | PT1| PX1| PT0| PX0|*/
                        /*|_________________|____________________|*/
     read24c02();       /*呼叫读取密码存放在 DATE1[]副程式*/
    dispbuf[0]=0x03;    /*小数点位*/
    dispbuf[7]=0x01;    /*ID 号码*/
    dispbuf[8]=0x01;    /*蜂鸣器设定开/二次分频*/
    Rd1302st();         /*读时钟芯片 DS1302*/
    LCD_disple();       /*液晶显示*/
    P0=0xff;
while(1)
    {
    scan();             /*呼叫键盘扫描副程式有按 FLAG0 会为 1*/
    if(FLAG0==1)
        {
```

```
            temp();                         /*判断有按键输入否？*/
            }
        else
            {FLAG4=FLAG5=0;
            if(FLAG2==1)
                {
                Rd1302();                   /*读时钟芯片 DS1302*/
                if(dispbuf[1]!=clockbak)    /*秒加一*/
                    {
                    clockbak=dispbuf[1];
                    tsled();                /*串行发送到 DTLED-6 副程式*/
                    tserial();              /*宣告串口 serial 发送副程式*/
                    }
                }
            }
        }
}
//************************************************************************/
//串行发送 DTLED-6 显示
//dispbuf0-dispbuf1-dispbuf2-dispbuf3-dispbuf4-dispbuf5-dispbuf6-dispbuf7-dispbuf8
//小数点 - 数码管 1- 数码管 2- 数码管 3- 数码管 4- 数码管 5- 数码管 6-ID 识别码-BZ 识别码
//00000xxx-0000xxxx-0000xxxx-0000xxxx-0000xxxx-0000xxxx-0000xxxx-0000xxxx-0000xxxx
//小数点=1~6,其它为不显示
//数码管 1-6=0~15，
//ID 识别码=0~15
//BZ 识别码=BZ.3 为蜂鸣输出控制位，0=关闭，1=打开
//      BZ0~2 为蜂鸣频率输出设定为，000=不分频，   最高频率输出=4800Hz；
//                                  001=一次分频，频率输出=4800/ 2 =2400Hz；
//                                  010=二次分频，频率输出=4800/ 4 =1200Hz；
//                                  011=三次分频，频率输出=4800/ 8 =600Hz；
//                                  100=四次分频，频率输出=4800/ 16=300Hz；
//                                  101=五次分频，频率输出=4800/ 32=150Hz；
//                                  110=六次分频，频率输出=4800/ 64=75Hz；
//                                  111=七次分频，频率输出=4800/128=38Hz；
//************************************************************************/
void tsled(void)            /*串行发送 DTLED-6 显示副程式*/
```

```
{     char ts0,i,j;
      clk=0;
      en=0;
      delay(10);
      for(j=0;j<9;j++)
      {     ts0=dispbuf[j];
            for(i=0;i<4;i++)        /*发小数点码 8，4，2，1*/
            {
            clk=1;
            ts0=ts0<<1;
            if(AC==1)
            dout=1;
            else          dout=0;
            clk=0;
            nop;
            delay(3);
            }
      }
      en=1;
}           /*返回主程式*/
/*************************************************************************/
//   设定  字符型 LCD 模块，使用 DB0--DB7,显示两行,
//   使用 5*7 字型    显示器要显示  光标要显示但不闪烁  */
//***********************************************************************/
void LCD_disple()                           /*液晶显示*/
{     delay(1000);
      Initial();
       Enable(0x80);                        /*  从第 1 行第 1 列开始显示  */
       Write1(char1);                       /*  显示第 1 行字符串  */
       Enable(0xc0);                        /*  从第 2 行第 1 列开始显示  */
       Write1(char2);                       /*  显示第 2 行字符串  */
       Enable(0xc9);                        /*  从第 2 行第 1 列开始显示  */
       Write3();                            /*  显示第 2 行字符串  */
}
//***********************************************************************/
void Initial()             /*LCD 模块初始化*/
```

```
{
        Enable(0x01);                  /*清除显示*/
       delay(300);                     /*清除需大于 1.64ms*/
        Enable(0x38);                  /*LCD 功能设定,8 位元数据传送,2 行显示*/
        Enable(0x0f);                  /*屏幕设定,游标 ON,游标闪烁*/
        Enable(0x06);                  /*加一状态,游标向右移*/
}
//*********************************************************************/
void Write1(char *c1)
{while(*c1!=0)
       {Write2(*c1);
       c1++;
       }
}
//*********************************************************************/
void Write3(void)
{uchar i=6;
while(i>0)
  {   P0=bufdata[i]|0x30;
       rs=1;rw=0;
       enable=1;
       delay(2);
       enable=0;
       i--;
  }
}
//*********************************************************************/
void Write2(char c2)
{P0=c2;
rs=1;rw=0;enable=1;
delay(2);
enable=0;
}
//*********************************************************************/
void Enable(uchar c)
{
```

```
P0=c;
rs=0;
rw=0;
enable=1;
delay(2);
enable=0;
}

/*********************************************************************/
void temp(void)
{     FLAG2=0;
   if(tab[ptr]==0x0e)
      {disp();                          /*如果是按“ C ”则呼叫显示密码 DISP*/
            while(m==P1);               /*按钮放开否？*/
        dispbuf[0]=0x03;                /*小数点位*/
      }
   else                                 /*否则往下执行*/
       {
       switch(tab[ptr])                 /*是则测试 ptr 键盘扫描计数器指标至 TAB[]取到的键盘码*/
             {
                case 0x0a:
                        write24c02();   /*是否按“ * ”是则呼叫设定密码 SET*/
                        clear();        /*是否按“ # ”是则呼叫清除显示器 CLEAR*/
                      while(m==P1);     /*按钮放开否？*/
                                 break; /*跳出此循环*/
                case 0x0b:
                        clear();        /*是否按“ # ”是则呼叫清除显示器 CLEAR*/
                      while(m==P1);     /*按钮放开否？*/
                                 break; /*跳出此循环*/
                case 0x0c:              /*是否按“ A ”，未规划键待用户自己定义*/
                        FLAG2=1; P3_2=1;
                        dispbuf[0]=0x03;      /*小数点位*/
                                 break; /*跳出此循环*/
                case 0x0d:              /*是否按“ B ”，未规划键待用户自己定义*/
                                 break; /*跳出此循环*/
                case 0x0e:              /*是否按“ C ”，已侦测过*/
```

```
                        read24c02(); /*呼叫读取密码存放在 DATE1[]副程式*/
                            break;   /*跳出此循环*/
                case 0x0f:
                        while(m==P1);                   /*按钮放开否？*/
                    open();             /*是否按“D”，是则呼比较密码开门副程式*/
                    dispbuf[0]=0x03;/*小数点位*/
                    goto openend;
                            break;   /*跳出此循环*/
                default:
                    xch();              /*以上均不是则为数字键呼叫 XCH 作右键滚入*/
                            break;   /*跳出此循环*/
                  }                     /*跳出 SWITCH*/
            delay(100);                 /*按钮抗机械反弹跳*/
            while(m==P1);               /*按钮放开否？*/
            tsled();                    /*串行发送到 DTLED-6 副程式*/
openend:      nop;
      }
}          /*返回上一层程式*/
//*******************************************************************/
void temp_setclock(void)                /*判断有按键输入否？*/
{
switch(dispbuf[0])                      /*是则测试 ptr 键盘扫描计数器指标至 TAB[]取到的键盘码*/
  {
   case 0x01:
                wxb1();                 /*0~9 计数*/
                break;                  /*跳出此循环*/
   case 0x03:
                wxb1();                 /*0~9 计数*/
                break;                  /*跳出此循环*/
   case 0x05:
                wxb2();                 /*0~2,0~4 计数*/
                break;                  /*跳出此循环*/
   default:                             /*以上均不是*/
                break;                  /*跳出此循环*/
  }
    delay(100);                         /*按钮抗机械反弹跳*/
```

```
    while(m==P1);                       /*按钮放开否？*/
    tsled();                            /*串行发送到 DTLED-6 副程式*/
}                                       /*返回上一层程式*/
//*****************************************************************/
void wxb1(void)
{
    switch(tab[ptr])                    /*是则测试 ptr 键盘扫描计数器指标至 TAB[]取到的键盘码*/
          {
            case 0x02:
                 dispbuf[dispbuf[0]]++;     /*是按“ 上 ”,是则呼叫设定密码 SET*/
                 if(dispbuf[dispbuf[0]]>9)
                     {dispbuf[dispbuf[0]+1]++;
                      dispbuf[dispbuf[0]]=0;
                     }
                 if(dispbuf[dispbuf[0]+1]==6)
                     {dispbuf[dispbuf[0]+1]=dispbuf[dispbuf[0]]=0;
                     }
                         break;             /*跳出此循环*/
            case 0x08:
                 if(dispbuf[dispbuf[0]]==0)/*是按“ 下 ”,是则呼叫清除显示器 CLEAR*/
                     {if(dispbuf[dispbuf[0]+1]==0)
                          dispbuf[dispbuf[0]+1]=6;
                          dispbuf[dispbuf[0]]=10;
                          dispbuf[dispbuf[0]+1]--;
                     }
                 dispbuf[dispbuf[0]]--;
                         break;             /*跳出此循环*/
            case 0x04:                      /*是按“ 左 ”，未规划键待用户自己定义*/
                 dispbuf[0]=dispbuf[0]+2;  /*小数点左移一位*/
                 if(dispbuf[0]>6)
                     dispbuf[0]=1;
                       break;               /*跳出此循环*/
            case 0x06:                      /*是按“ 右 ”，未规划键待用户自己定义*/
                 if(dispbuf[0]<3)
                     dispbuf[0]=7;
                 dispbuf[0]=dispbuf[0]-2;  /*小数点右移一位*/
```

```
                break;              /*跳出此循环*/
        case 0x0a:                  /*是按“＊”，已侦测过*/
                FLAG5=1;
                  break;            /*跳出此循环*/
        case 0x0c:                  /*是否按“A”，未规划键待用户自己定义*/
            FLAG4=FLAG5=P3_2=1;
            dispbuf[0]=0x03;        /*小数点位*/
                 break;             /*跳出此循环*/
        default:                    /*以上均不是*/
                break;              /*跳出此循环*/
       }
}                                   /*返回上一层程式*/
//*******************************************************************/
void wxb2(void)
{switch(tab[ptr])               /*是则测试 ptr 键盘扫描计数器指标至 TAB[]取到的键盘码*/
       {case 0x02:
            dispbuf[dispbuf[0]]++;  /*是按“上”,是则呼叫设定密码 SET*/
            if(dispbuf[dispbuf[0]+1]==2)
                {if(dispbuf[dispbuf[0]]>4)
                    {dispbuf[dispbuf[0]+1]=dispbuf[dispbuf[0]]=0;}
                }
            else
                {if(dispbuf[dispbuf[0]]>9)
                    {dispbuf[dispbuf[0]+1]++;   dispbuf[dispbuf[0]]=0;}
                if(dispbuf[dispbuf[0]+1]>2)
                    {dispbuf[dispbuf[0]+1]=dispbuf[dispbuf[0]]=0;}
                }
                break;              /*跳出此循环*/
        case 0x08:
            if(dispbuf[dispbuf[0]]==0)   /*是按“下”,是则呼叫清除显示器 CLEAR*/
                {if(dispbuf[dispbuf[0]+1]==0)
                    {dispbuf[dispbuf[0]+1]=2;
                     dispbuf[dispbuf[0]]=5;
                    }
                else
                    {dispbuf[dispbuf[0]]=10; dispbuf[dispbuf[0]+1]--;
```

```
                        }
                    }
                    dispbuf[dispbuf[0]]--;
                        break;              /*跳出此循环*/
        case 0x04:                          /*是按“ 左 ”，未规划键待用户自己定义*/
            dispbuf[0]=dispbuf[0]+2;        /*小数点左移一位*/
            if(dispbuf[0]>6)
                dispbuf[0]=1;
                  break;                    /*跳出此循环*/
        case 0x06:                          /*是按“ 右 ”，未规划键待用户自己定义*/
            if(dispbuf[0]<3)
                dispbuf[0]=7;
            dispbuf[0]=dispbuf[0]-2;        /*小数点右移一位*/
                        break;              /*跳出此循环*/
        case 0x0a:                          /*是按“ * ”，已侦测过*/
                        FLAG5=1;
                          break;            /*跳出此循环*/
        case 0x0c:                          /*是否按“ A ”，未规划键待用户自己定义*/
            FLAG4=FLAG5=P3_2=1;
            dispbuf[0]=0x03;                /*小数点位*/
                          break;            /*跳出此循环*/
        default:                            /*以上均不是*/
                          break;            /*跳出此循环*/
      }
}                                           /*返回上一层程式*/
//*******************************************************************/
void save_clocks(void)                      /*按显示时间更改时钟*/
{clocktmp[0]=dispbuf[1]|(dispbuf[2]<<4);
 clocktmp[1]=dispbuf[3]|(dispbuf[4]<<4);
 clocktmp[2]=dispbuf[5]|(dispbuf[6]<<4);
 Wr1302();
 FLAG2=1;
 nop;
}
//*******************************************************************/
void xch(void)                              /*显示向右移动副程式*/
```

```
{char c;                                  /*宣告变数*/
     for(c=6;c>0;c--)
     {dispbuf[c]=dispbuf[c-1];
                      /*将显示器 dispbuf[]移位交换，dispbuf[3]->dispbuf[4]->dispbuf[5]...*/
     }
     dispbuf[0]=0;                        /*有安键时,小数点不亮*/
     dispbuf[1]=tab[ptr];                 /*新的按键值存入 dispbuf[1],最低位*/
}                                         /*返回上一层程式*/
//***************************************************************/
void open(void)                           /*开门比较密码副程式*/
{char c;                                  /*宣告变数*/
   FLAG1=1;                               /*设开门标号为 1*/
for(c=1;c<6;c++)                          /*比较六个密码*/
 {if(dispbuf[c]!=bufdata[c])              /*显示值(按键值 disbuf[])与密码 TAB[]比较是否相等*/
        FLAG1=0;
         if(FLAG1==0)                     /*FLAG1=0  表示不相等*/
               break;                     /*跳出此循环*/
 }
 if(FLAG1==1)                             /*如 FLAG1=1  表示密码相等,令电锁 P2.0 置 0 动作*/
  {                                       /*密码正确*/
    dispbuf[8]=0x09;                      /*蜂鸣器设定开/二次分频*/
   dispbuf[0]=0x01;                       /*小数点到最高位位*/
   P3_2=0;                                /*P3.2 置 0,使绿色下载指示灯亮(喇叭响)*/
//   TMOD=0x11;
   TR1=1;                      /**/
   TF1=0;
    IE=0x9f;                         /*8 是允许中断，a 是 TIMER0，TIMER1 中断致能位*/
   tsled();                          /*串行发送到 DTLED-6 副程式*/
   while(FLAG5==0)                   /*等待设置死循环*/
       {
        scan();                      /*呼叫键盘扫描副程式有按 FLAG0 会为 1*/
        if(FLAG0==1)
         {
          temp_setclock();           /*判断有按键输入否？*/
         }
        if(FLAG6==1)
```

```
                {tsled();                         /*串行发送到 DTLED-6 副程式*/
                FLAG6=0;
                }
           }
      if(FLAG4==0)
       save_clocks();                             /*按显示时间更改时钟*/
   }
       else P3_2=1;                               /*密码错，P3.2 置 1,关闭绿色下载指示灯灭(喇叭不响)*/
}                                                 /*返回上一层程式*/
//*********************************************************************/
void clear(void)                                  /*清除显示器 dispbuf[]为 00 副程式*/
{char c;                                          /*宣告变数*/
        for(c=1;c<7;c++)                          /*将 00 存入显示阵列 dispbuf[]*/
        {dispbuf[c]=0x00;
        }
}                                                 /*返回上一层程式*/
//*********************************************************************/
void Wr1302(void)                                 /*写时钟芯片 DS1302*/
{char i,j,ts0,conb,conbyt;
      conb=0x80;
      ts0=conb;
//
for(j=0;j<3;j++)                                  /*写秒,分,时三次循环*/
      {clk=RST=0;                                 /*DS1302 系统初始化*/
           nop;nop;
           RST=1;
      for(i=0;i<8;i++)                            /*写控制字到 DS1302*/
         {clk=0;                                  /*清时钟*/
           if(conb&0x01)
                      DINOUT=1;
           else       DINOUT=0;
           clk=1;                                 /*时钟上升沿，发送数据有效*/
           conb=conb>>1;
           nop;
         }
           clk=0;                                 /*清时钟*/
```

```
        conb=ts0+2+j*2;
    conbyt=clocktmp[j]&0x7f;
    for(i=0;i<8;i++)                /*写设定值到 DS1302*/
      {clk=0;                       /*清时钟*/
       if(conbyt&0x01)
                DINOUT=1;
        else    DINOUT=0;
        clk=1;                      /*时钟上升沿，发送数据有效*/
        conbyt=conbyt>>1;
        nop;
      }
    delay(5);
    RST=0;
nop;
}
    }
                                    /*返回主程式*/
//*****************************************************************************/
void Rd1302st()                     /*启动时钟芯片振荡器*/
{
char i,ts0,conb,conbyt;
    Rd1302();
    conbyt=clocktmp[0]&0x7f;
    conb=0x80;
    ts0=conb;
    clk=RST=0;                      /*DS1302 系统初始化*/
    nop;nop;
    RST=1;
    for(i=0;i<8;i++)                /*写控制字到 DS1302*/
      { clk=0;                      /*清时钟*/
        if(conb&0x01)
                DINOUT=1;
        else    DINOUT=0;
        clk=1;                      /*时钟上升沿，发送数据有效*/
        conb=conb>>1;
        nop;
```

```
    }
      clk=0;                  /*清时钟*/
      for(i=0;i<8;i++)        /*写控制字到 DS1302*/
    { clk=0;                  /*清时钟*/
      if(conbyt&0x01)
              DINOUT=1;
      else    DINOUT=0;
      clk=1;                  /*时钟上升沿，发送数据有效*/
      conbyt=conbyt>>1;
      nop;
    }
  delay(5);
  RST=0;
nop;
}       /*返回主程式*/
//***********************************************************************/
void Rd1302()                 /*读时钟芯片 DS1302*/
{char i,j,ts0,conb;
  conb=0x81;
  clocktmp[0]=0;
  clocktmp[1]=0;
  clocktmp[2]=0;
  ts0=conb;
for(j=0;j<3;j++)
  {clk=RST=0;                 /*DS1302 系统初始化*/
  nop;nop;
  RST=1;
  for(i=0;i<8;i++)            /*写控制字到 DS1302*/
    {
      clk=0;                  /*清时钟*/
      if(conb&0x01)
              DINOUT=1;
      else    DINOUT=0;
      clk=1;                  /*时钟上升沿，发送数据有效*/
      conb=conb>>1;
      nop;
```

```
        }
      conb=ts0=ts0+2;
      clk=0;
      for(i=0;i<7;i++)                    /*读时钟芯片的时分秒单元*/
      {
      if(DINOUT==1)
              clocktmp[j]=clocktmp[j]|0x80;
      else    clocktmp[j]=clocktmp[j]&0x7f;
      clk=1;
      clocktmp[j]=clocktmp[j]>>1;         /*接收暂存单元*/
      nop;
      clk=0;                              /*时钟下降沿，接收数据有效*/
      }
      delay(5);
      RST=0;
      delay(10);
     }
      dispbuf[1]=clocktmp[0]&0x0f;        /*转换秒分时显示单元*/
      dispbuf[2]=clocktmp[0]>>4&0x07;
      dispbuf[3]=clocktmp[1]&0x0f;
      dispbuf[4]=clocktmp[1]>>4&0x07;
      dispbuf[5]=clocktmp[2]&0x0f;
      dispbuf[6]=clocktmp[2]>>4&0x03;
      nop;
}                                         /*返回主程式*/
//*******************************************************************/
void read24c02()                          /*读 24c02 的存储值*/
{uchar i;
     TRcvAddress(0xa0,0,&bufdata[0],7);
     for(i=0;i<6;i++)
       {dispbuf[i]=bufdata[i];
       }
}                                         /*返回上一层程式*/
//*******************************************************************/
void write24c02()                         /*写 24c02 的存储值*/
{TSendAddress(0xa0,0,dispbuf,7);          /* 向有子地址器件写入 6 字节数据函数 */
```

```
}
//*************************************************************************/
void tserial2(void)                    /*宣告串口 serial 发送副程式*/
{ uchar i=7;
IE=0x8f;
   {while(i>1)
      {
      i--;
      SBUF=dispbuf[i];                 /*根据扫描指标到 TAB 中取 ASCII 码由 SBUF 发送出去*/
      while(TI!=1);                    /*发送完成否？*/
      TI=0;                            /*是，则清除发送完成旗标 TI=0*/
      delay(10);
      }
   }
 IE=0x9f;
}
//*************************************************************************/
void tserial(void)                     /*宣告串口 serial 发送副程式*/
{ uchar sec,min,hour;
      sec=((dispbuf[2]<<4)&0xf0)|(dispbuf[1]&0x0f);
      min=((dispbuf[4]<<4)&0xf0)|(dispbuf[3]&0x0f);
      hour=((dispbuf[6]<<4)&0xf0)|(dispbuf[5]&0x0f);
 IE=0x8f;
      SBUF=hour;                       /*根据扫描指标到 TAB 中取 ASCII 码由 SBUF 发送出去*/
      while(TI!=1);                    /*发送完成否？*/
      TI=0;                            /*是，则清除发送完成旗标 TI=0*/
      delay(20);
      SBUF=min;                        /*根据扫描指标到 TAB 中取 ASCII 码由 SBUF 发送出去*/
      while(TI!=1);                    /*发送完成否？*/
      TI=0;                            /*是，则清除发送完成旗标 TI=0*/
      delay(20);
      SBUF=sec;                        /*根据扫描指标到 TAB 中取 ASCII 码由 SBUF 发送出去*/
      while(TI!=1);                    /*发送完成否？*/
      TI=0;                            /*是，则清除发送完成旗标 TI=0*/
      delay(20);
//    IE=0x9f;
```

```
}
//**********************************************************************/
//      24C02 芯片读写                    起动总线函数
//函数原型: void    Start();
//功能:          启动总线,发送 24c02 启动条件.
//**********************************************************************/
void Start()
{SDA=0;
nop;
nop;
     SCL=0;
     SDA=1;              /*发送起始条件的数据信号*/
     nop;
     SCL=1;
     nop;                /*起始条件建立时间大于 4.7us,延时*/
     nop;
     nop;
     nop;
     nop;
     SDA=0;              /*发送起始信号*/
     nop;                /*起始条件锁定时间大于 4us*/
     nop;
     nop;
     nop;
     nop;
     SCL=0;              /*钳住总线，准备发送或接收数据  */
     nop;
     nop;
     SDA=1;
}
//**********************************************************************/
//                                  结束总线函数
//函数原型: void    Stop();
//功能:          结束总线,发送 24c02 结束条件.
//**********************************************************************/
void Stop()
```

```
{
   SCL=0;
   SDA=0;                    /*发送结束条件的数据信号*/
   nop;                      /*发送结束条件的时钟信号*/
   SCL=1;                    /*结束条件建立时间大于 4us*/
   nop;
   nop;
   nop;
   nop;
   nop;
   SDA=1;                    /*发送总线结束信号*/
   nop;
   nop;
   nop;
   nop;
   SCL=0;
}
//******************************************************************/
//                    89X52 发数据或地址给 24c02  字节数据发送函数
//函数原型: void    SendByte(uchar c);
//功能:      将数据 c 发送出去,可以是地址,也可以是数据,发完后等待应答,并对
//           此状态位进行操作.(不应答或非应答都使 acknow=0)
//           发送数据正常，acknow=1; acknow=0 表示被控器无应答或损坏。
//******************************************************************/
void    SendByte(uchar c)
{ uchar count;
for(count=0;count<8;count++)       /*要传送的数据长度为 8 位*/
        {if((c<<count)&0x80)
          SDA=1;                     /*判断发送位*/
         else
          SDA=0;
         nop;
         SCL=1;                     /*置时钟线为高，通知被控器开始接收数据位*/
          nop;
          nop;                      /*保证时钟高电平周期大于 4us*/
          nop;
```

```
          nop;
          nop;
        SCL=0;
      }
      nop;
      nop;
      SDA=1;                    /*8 位发送完后释放数据线，准备接收应答位*/
      nop;
      nop;
      SCL=1;
      nop;
      nop;
      nop;
      if(SDA==1)
         acknow=0;              /* 24c02 无应答  */
      else
         acknow=1;              /*  发送数据正常  */
      SCL=0;
      nop;
      nop;
}
//************************************************************************/
//                    89X52 从 24c02 读数据字节数据接收函数
//函数原型: uchar    RcvByte();
//功能:         用来接收从器件传来的数据,并判断总线错误(不发应答信号)，
//              发完后请用应答函数应答从机。
//************************************************************************/
uchar    RcvByte()
{uchar retc=0;
 uchar count;
SDA=1;                          /*置数据线为输入方式*/
for(count=0;count<8;count++)
    {      nop;
           SCL=0;               /*置时钟线为低，准备接收数据位*/
           nop;
           nop;                 /*时钟低电平周期大于 4.7us*/
```

```
            nop;
            nop;
            nop;
            SCL=1;                  /*置时钟线为高使数据线上数据有效*/
            nop;
            nop;
            retc=retc<<1;
            if(SDA==1)
            retc=retc+1;            /*读数据位,接收的数据位放入 retc 中  */
            nop;
            nop;
    }
    SCL=0;
    nop;
    nop;
    return(retc);
}
//********************************************************************/
//                         主机 89X52 应答子函数
//函数原型:    void Ack(bit a);
//功能:        主控器进行应答信号(可以是应答或非应答信号，由位参数 a 决定)
//********************************************************************/
void Ack(bit a)
{if(a==0)
        SDA=0;                  /*在此发出应答或非应答信号  */
    else                        /*  应答  a=0    非应答  a=1 */
        SDA=1;
    nop;
    nop;
    nop;
    SCL=1;
    nop;
    nop;                        /*时钟低电平周期大于 4us*/
    nop;
    nop;
    nop;
```

```
   SCL=0;                              /*清时钟线，钳住 I2C 总线以便继续接收*/
   nop;
   nop;
}
//*******************************************************************/
//                          用户接口函数
// 有无子地址表示是否向芯片的特定地址写数据
//                向有子地址器件写入 6 字节数据函数    对应数据资料中的写模式
//函数原型: bit    TSendAddress(uchar slaaddress,uchar subaddress,uchar * s,uchar no);
//功能:          从启动总线到发送从地址，子地址,数据，结束总线的全过程,从器件
//                地址 slaaddress，子地址 subaddress，发送内容是 s 指向的内容,no=字节数
//                如果返回 1 表示操作成功，否则操作有误。
//注意：          使用前必须已结束总线。
//*******************************************************************/
bit TSendAddress(uchar slaaddress,uchar subaddress,uchar * s,uchar no) /* 向有子地址器件写入 6 字节数据函数 */
{uchar i;
     Start();                          /*启动总线*/
     SendByte(slaaddress);             /*发送器件地址*/
     if(acknow==0)
     return(0);
     SendByte(subaddress);             /*发送器件子地址*/
     if(acknow==0)
     return(0);
     for(i=0;i<no;i++)
     {
        SendByte(s[i]);                /*发送数据*/
        if(acknow==0)
        return(0);
     }
     Stop();                           /*结束总线*/
     return(0);
}
//==================================================================
//                               向有子地址器件读取 4 字节数据函数
//函数原型: bit    TRcvAddress(uchar slaaddress,uchar subaddress,uchar * s,uchar no);
```

```
//功能:        从启动总线到发送从地址，子地址,读数据，结束总线的全过程,从器件
//              地址 slaaddress，子地址 subaddress，读出的内容放入 s, no=字节数
//               如果返回 1 表示操作成功，否则操作有误。
//注意：        使用前必须已结束总线。
//========================================================================
bit TRcvAddress(uchar slaaddress,uchar subaddress,uchar * s,uchar no)
{ uchar i;
    Start();                                    /*启动总线*/
    SendByte(slaaddress);                       /*发送器件从地址*/
    if(acknow==0)
    return(0);
    SendByte(subaddress);                       /*发送器件子地址*/
    if(acknow==0)
    return(0);
    Stop();                                   /*结束总线*/
    Start();                                  /*重新启动总线*/
    SendByte(slaaddress+1);
    if(acknow==0)
    return(0);
    for(i=0;i<no;i++)
     {* s=RcvByte();                          /*接收数据*/
       Ack(0);                                /*发送应答位*/
       s++;
     }
       * s=RcvByte();                           /*接收数据*/
       Ack(1);
       Stop();                                  /*结束总线*/
       return(1);
}
//========================================================================
void disp(void)                               /*显示密码副程式*/
{     char c;                                 /*宣告变数*/
          read24c02();                        /*呼叫读取密码存放在 DATE1[]副程式*/
      ptr=0xff;
           for(c=1;c<7;c++)               /*将密码存放在阵列 buft[]存入显示器存放阵列 dispbuf[]*/
         {
```

```
        dispbuf[c]=bufdata[c];
       }
         tsled();                        /*串行发送到 DTLED-6 副程式*/
         while(m==P0);                   /*判断按钮放开否，没有则在此等待*/
         clear();                        /*有则呼叫清除显示器*/
        FLAG2=1;
}
//*********************************************************************
void scan(void)                          /*扫描键盘副程式*/
{char a1=0xef,i;                         /*A1=0XF7 列扫描初值，I 行*/
         FLAG0=0;                        /*设按键回应旗号为 0，键盘扫描计数指标为 0*/
         ptr=0;
         for(i=0;i<4;i++)                /*键盘 4 个扫描列*/
       {
            P1=a1;        /*列扫描输出，读入 P1 存入 M，以便侦测行与侦测按键是否放开*/
            m=P1;
            switch(m&0x0f)               /*取行的高 4 位元，侦测那一行被按*/
             {
               case 0x07: ptr=i*4;                /*第一行被按否？是则扫描指标=列 X4*/
                            FLAG0=1;      /*是则设 FLAG0=1 表有按键输入*/
                            break;        /*跳出此循环*/
                 case 0x0b: ptr=i*4+1;            /*第二行被按否？是则扫描指标=列 X4+1*/
                            FLAG0=1;      /*是则设 FLAG0=1 表有按键输入*/
                            break;        /*跳出此循环*/
                 case 0x0d: ptr=i*4+2;            /*第三行被按否？是则扫描指标=列 X4+2*/
                            FLAG0=1;      /*是则设 FLAG0=1 表有按键输入*/
                            break;        /*跳出此循环*/
                 case 0x0e: ptr=i*4+3;            /*第四行被按否？是则扫描指标=列 X4+3*/
                            FLAG0=1;      /*是则设 FLAG0=1 表有按键输入*/
                  default:    break;      /*跳出此循环*/
             }
          if(FLAG0==1)break;              /*不为 1，则扫描列右移，扫描下一列*/
          a1=a1<<1|0x01;     /*高位补 1，由于 P1.7~P1.4 未接+5V，而是由指令加载高电平*/
        }
        //P1=0xff;
 }                                        /*返回主程式*/
```

```
/*******************************************************/
void service_int1 () interrupt 3 using 3        /*TIMER1 中断副程式--秒计时器*/
{ TH1=(65525-59200)/256;                        /*重设 TIMER1 计数值*/
   TL1=(65525-59200)%256;
   dispbuf[8]=0x09;                             /*蜂鸣器设定开/二次分频*/
   if(b1==0)                                    /*中断次数完成否，是则表 1 秒到了*/
      {
        b1=100;                                 /*重设中断次数*/
        P3_2=1;                                 /*秒 SEC 加 1*/
     TR1=0;
     dispbuf[8]=0x01;                           /*蜂鸣器关*/
     clear();
     tsled();                                   /*串行发送到 DTLED-6 副程式*/
      }
   TF1=0;
   b1--;                                        /*中断次数减 1*/
}                                               /*返回主程式*/
/****************************************************************/
void uartcom1_in1 () interrupt 4 using 1        /*串行口中断副程式*/
 {   uchar v,i=0;
        while(RI!=1);                           /*接收完成否*/
         {
     RI=0;
     v=SBUF;                                    /*接收的数据存入 C*/
     dispbuf[6]=dispbuf[5];
     dispbuf[5]=dispbuf[4];
     dispbuf[4]=dispbuf[3];
     dispbuf[3]=dispbuf[2];
     dispbuf[2]=dispbuf[1];
     dispbuf[1]=SBUF;
     FLAG6=1;
         }
 }                                              /*返回主程式*/
/****************************************************************/
```

４—2．电路原理图

请见附录（一）

４—3．结构示意图

请见附录（二）

４－4．包装清单

a．　主机板　　　　　　　＊1 块

b．　使用说明书　　　　　＊1 本

c．　软件光盘　　　　　　＊1 张（包含软件）

a)　SY_1126.C 的源程序

b)　SY_1126 板电原理图

c)　UltraEdit 编辑软件

d)　A51,C51 的编译连接软件

e)　CordCruiser51 仿真软件

f)　Winisp 下载软件

g)　Atmel AVR Isp 下载软件

h)　SLISP 并口下载软件

i)　串口调试检测软件

第五章：赛博其它 51 类实验板简介

5－1．51DEMO I/O 板简介

51XXDEMO 板结构简单，使用方便，是检测仿真器端口的最佳工具板。51XXDEMO 板的 P0，P1，P2，P3 各口已直接联接上四个七段 LEO 数码管显示模块，并配有演示示范程序，将板上单片机取下，插上 51 系列仿真器的仿真头后，可以直接监测到仿真器 P0，P1，P2，P3 端口的输出状态。是用户用来检测仿真器输出端口的最好工具，板上并扩展了液晶显示的模式。适用于初学者及数学试验和直接用于产品开发。此产品还提供了方便实用的双电源供电插座，DC+5V 或 AC/8-12V 电源均可。祥情请见该产品的资料说明。

5－2．A/D-8X51 数模转换开发实验板简介

A/D-8X51 模数转换板像其它 51 类的开发实验板一样，上电复位后进入启动工作状态，程序自动的控制 AD0804 模数转换器，将 0—5 伏的模拟电压转换成数字电压，供 8X51 单片机处理后送 LED 数码管显示；在本案中是采用了 AD590 的温度传感器将变化的温度转换成变化的电压，再将电压的变化值转换成对应的温度显示出来；如测到的温度在设定的上下限之间，绿色的正常指示灯亮，反之温度大于或小于设定温度的上下限，红色（HI/LOW）指示灯将分别闪亮，表示报警。

板上的配制有：

a． 标准的 8X51 应用电路设计。

b． 带 4 位数码管显示 LED，可显示 3 位数电压或转换成的温度。

c． 自设 4 个功能设置按键，或用于上下（加减）调整之

用。在板上分别为是 MODE，SET，UP，DOWN。

d． 有方便实用的双电源供电插座，DC+5V 或 AC9V-12V 电源均可。

e． 有三路 LED 显示的控制输出指示，并可直接在板上再扩展输出控制。

f． 预留 8X51 所有端口的连接插口，便于用户二次开发成成品。

祥情请见该产品的资料说明（型号为：SY-1032）。

5－3．流水灯控制器（12 路）

外型如下图：

流水灯控制器是用 89C2051 单片机为机芯，可直接使用于广告灯的循环时序控制以及工业现场控制等。将控制器接上电源 DC-+5V 或 DC-+12V 后，面板上侧的绿色指示灯亮，12 路指示灯自动会象流水似的移动闪烁；如需提高或降低流水指示灯的变换速度，只需按住加速（减速）钮，这时可以看到指示灯在做二进制的减法（加法），松开按钮后指示灯的变换速度则会

有相应的变化；按住正/反向钮一次，则会自动变换一次流水灯的循环方向一次。

5－4．ISP 下载线（选配自购件）

专用下载器(正面)

专用下载器(反面)

将 AVR 下载线 25pin 接口端连接在 PC 机的并口（打印口）上，10pin 排线端通过排线插在本开发板上的 “ ISP ” 插座上，开发板接上+5 伏电源，下载线上有红色指示灯亮即可启动下载软件进行下载（烧录）AT89S8252 芯片。

请选购由深圳得技通研制的“ **AVR** 下载线 ”。特供配套价每只 **45.00** 元。

**********公司其它产品简介见软件盘中电子版文件**********